DynFrs: An Efficient Framework for Machine Unlearning in Random Forest

Shurong Wang[1] *Zhuoyang Shen*[1] *Xinbao Qiao*[1]
Tongning Zhang[1] *Meng Zhang*[1]*
[1]Zhejiang University *Corresponding Author

The Thirteenth International Conference on Learning Representations
ICLR 2025

Overview

Machine Unlearning
Efficiently remove the influence of specific sample(s) from the model (entirely or approximately)

Random Forest
An ensemble learning method that builds multiple Decision Trees for better accuracy and robustness.

Functionality Check

	Sequential Insertion	Batch Insertion	Sequential Removal	Batch Removal	Exactness	Inference
Random Forest	✗	✗	✗	✗	✓	✓
DaRE	✗	✗	✓	✗	✓	✓
HedgeCut	✗	✗	✓	✗	✓	✓
OnlineBoosting	✗	✗	✗	✓	✗	✓
DynFrs (ours)	✓	✓	✓	✓	✓	✓

Sequential/Batch: one/multiple sample(s) in one request.
Insertion: add new sample to the model (Continual Learning)
Removal: delete sample from the model (Machine Unlearning)
Exactness: the impact of the deleted sample is removed entirely

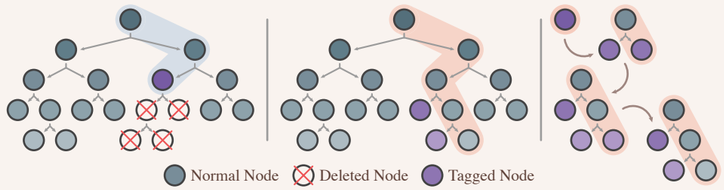
Our contributions
- Enable *real-time low-latency* sample insertion/removal for Random Forest.
- Our framework demonstrates *strong empirical performance* and *theoretical soundness*.
- Narrow the gap between traditional machine learning models and complicated privacy-sensitive real-world scenarios.

Passionate Self-Motivated Undergrad Looking for PhD Position.
Shurong Wang, email: shurong.22@intl.zju.edu.cn

Methods

Extremely Randomized Trees
We adopt modified ERTs as the forest's base learners, which is originally proposed by *Geurts* et al. (2006).
Split decisions in ERTs are less dependent on training samples, leading to *more robustness* against sample insertion/removal. ERTs enable more *efficient* continual learning and machine unlearning in DynFrs while retaining **competitive accuracy**.

Lazy Tag Strategy (LZY)
Turn subtree reconstruction into ***path reconstruction*** in queries. Find the optimal node-level modification batching strategy by placing lazy tags.



Left: (a) A sample addition/removal request arises, and its impacts are covered in the blue path. (b) A change in best split happens in the purple node, so a tag is placed on it. (c) The subtree of the tagged node is deleted.
Middle: (a) A querying request arises, and the nodes it visits are marked as the orange path. (b) The tag is pushed down recursively until the query reaches a leaf.
Right: A detailed process of how the querying request in *Middle* grows a tree path until a leaf is reached.

Tree Subsampling (OCC(q))
For each sample, do a subsampling on trees. Limits the occurrence to $k = \lceil qT \rceil$ trees only (for some $q < 1$), reducing the workload in training and sample insertion/removal.

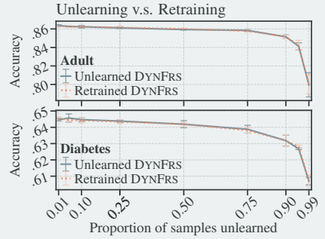
OCC(q) reduces workload in sample insertion/deletion by a factor of $1/q^2$ and $1/q$ in training compared to naive retraining. Therefore, a ***100 or 25 times speedup*** is offered for sample insertion/deletion for $q = 0.1$ or 0.2 as suggested.

Results

Predictive Performance
The accuracy of the model on prediction tasks, where DynFrs stands out.

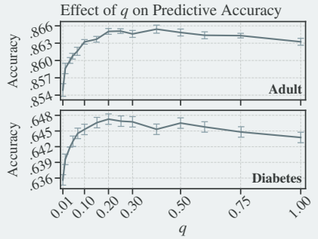
Datasets	DaRE	HedgeCut	Vanilla	Online Boosting	DynFrs (q = 0 .1)	DynFrs (q = 0 .2)
Purchase	.9327±.001	.9118±.001	**.9372**±.001	.9207±.000	.9327±.001	.9359±.001
Vaccine	.7916±.003	.7706±.002	**.8012**±.000	.7939±.002	.7911±.001	.7934±.002
Adult	.8628±.001	.8428±.001	.8637±.000	.8503±.000	.8633±.001	**.8650**±.001
Bank	.9420±.000	.9350±.000	.9414±.001	**.9436**±.000	.9417±.000	**.9436**±.000
Heart	.7344±.001	.7195±.001	.7342±.001	.7301±.000	.7358±.002	**.7366**±.000
Diabetes	.6443±.001	.6190±.000	.6435±.001	.6462±.000	.6453±.001	**.6470**±.002
NoShow	.7361±.001	.7170±.000	**.7387**±.000	.7269±.000	.7335±.001	.7356±.000
Synthetic	.9451±.000	/	.9441±.000	.9309±.000	.9424±.000	**.9454**±.000
Higgs	.7441±.000	/	.7434±.000	.7255±.000	.7431±.000	**.7475**±.000

Unlearning Exactness
DynFrs can remove all the impacts of the removed sample(s) have had on the model

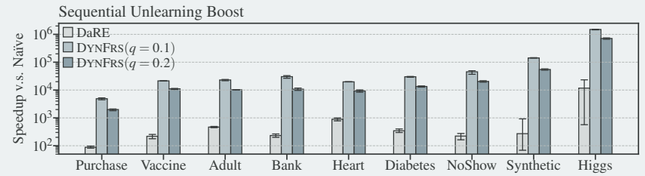


Effects of Hyperparameter q
Predictive Performance of DynFrs on different q.

$q = 0.1$ is a good option for unlearning efficency, while $q = 0.2$ is great for accuracy



Sequential Unlearning Efficiency
The speedup provided by the model in machine unlearning compared to naive retraining. Larger is better.
DynFrs easily outperforms other models in this setting.



Batch Unlearning Efficiency
The time used to unlearn a batch of samples at the same time. Smaller is better.

Clearly, DynFrs is the fastest in this setting, and is the only model that converges for extremely large batch.